Welcome to SQL Server Integration Services

SQL Server Integration Services (SSIS) is one of the most powerful features in SQL Server 2005. It is technically classified as a business intelligence feature and is a robust way to load data and perform tasks in a workflow. Even though it's mainly used for data loads, you can use it to do other tasks in a workflow like executing a program or a script, or it can be extended. This chapter describes much of the architecture of SSIS and covers the basics of tasks.

What's New in SQL Server 2005 SSIS

In SQL Server 7.0, Microsoft had a small team of developers work on a much understated feature of SQL Server called Data Transformation Services (DTS). DTS was the backbone of the Import/Export Wizard, and the DTS's primary purpose was to transform data from almost any OLE DB–compliant data source to another destination. It also had the ability to execute programs and run scripts, making workflow a minor feature.

By the time that SQL Server 2000 was released, DTS had a strong following of DBAs and developers. Microsoft included in the release new features like the Dynamic Properties task to help you dynamically alter the package at runtime. It also had extended logging and broke a transformation into many phases, called the multiphase data pump. Usability studies still showed that at this point developers had to create elaborate scripts to extend DTS to do what they wanted. For example, if you wanted DTS to conditionally load data based on the existence of a file, you would have to use the ActiveX Script task and VBScript to dynamically do this. The problem here was that most DBAs didn't have this type of scripting experience.

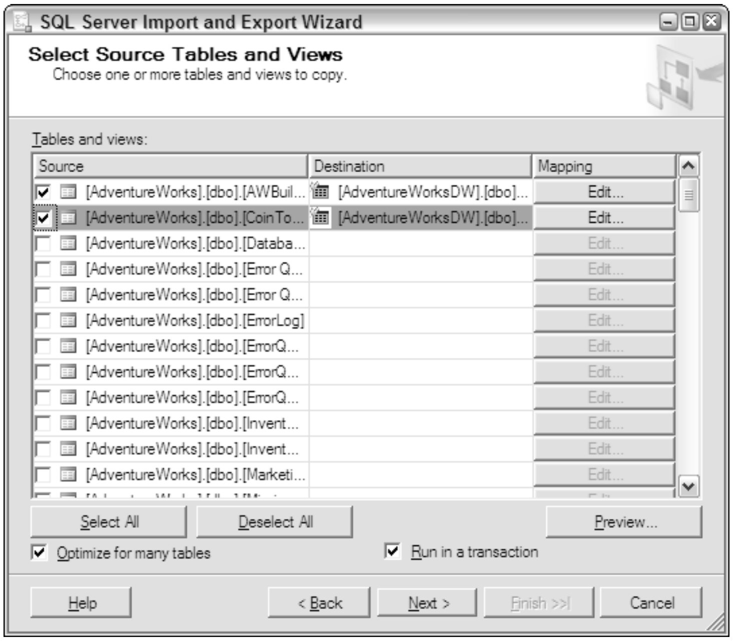
After five years, Microsoft released the much touted SQL Server 2005, where DTS is no longer an understated feature, but one of the main business intelligence (BI) foundations. It's been given so much importance now that it has its own service. DTS has also been renamed to SQL Server Integration Services (SSIS). So much has been added to SSIS that the rename of the product was most appropriate. Microsoft made a huge investment in usability and making it so that there is no longer a need for scripting.

Most of this book will assume that you know nothing about the past releases of SQL Server DTS and will start with a fresh look at SQL Server 2005 SSIS. After all, when you dive into the new features, you'll realize how little knowing anything about the old release actually helps you when learning this one. The learning curve can be considered steep at first, but once you figure out the basics, you'll be creating what would have been complex packages in SQL Server 2000 in minutes.

You can start differentiating the new SSIS by looking at the toolbox that you now have at your fingertips as an SSIS developer. The names of the tools and how you use them have changed dramatically, but the tools all existed in a different form in SQL Server 2000. This section introduces you briefly to each of the tools, but you will explore them more deeply beginning in the next chapter.

Import and Export Wizard

If you need to move data quickly from almost any OLE DB–compliant data source to a destination, you can use the SSIS Import and Export Wizard (shown in Figure 1-1). The wizard is a quick way to move the data and perform very light transformations of data. It has not changed substantially from SQL Server 2000. Like SQL Server 2000, it still gives you the option of checking all the tables you'd like to transfer. You also get the option now of encapsulating the entire transfer of data into a single transaction.


SQL Server Import and Export Wizard
Select Source Tables and Views
Choose one or more tables and views to copy.
Tables and views:

Source	Destination	Mapping
✓ [AdventureWorks].[dbo].[AWBuil...	[AdventureWorksDW].[dbo]...	Edit...
✓ [AdventureWorks].[dbo].[CoinTo...	[AdventureWorksDW].[dbo]...	Edit...
[AdventureWorks].[dbo].[Databa...		Edit...
[AdventureWorks].[dbo].[Error Q...		Edit...
[AdventureWorks].[dbo].[Error Q...		Edit...
[AdventureWorks].[dbo].[ErrorLog]		Edit...
[AdventureWorks].[dbo].[ErrorQ...		Edit...
[AdventureWorks].[dbo].[ErrorQ...		Edit...
[AdventureWorks].[dbo].[ErrorQ...		Edit...
[AdventureWorks].[dbo].[Invent...		Edit...
[AdventureWorks].[dbo].[Invent...		Edit...
[AdventureWorks].[dbo].[Marketi...		Edit...

✓ Optimize for many tables

✓ Run in a transaction

Figure 1-1

The Business Intelligence Development Studio

The Business Intelligence Development Studio (BIDS) is the central tool that you'll spend most of your time in as a SQL Server 2005 SSIS developer. Like the rest of SQL Server 2005, the tool's foundation is the Visual Studio 2005 interface (shown in Figure 1-2), which is the equivalent of the DTS Designer in SQL Server 2000. The nicest thing about the tool is that it's not bound to any particular SQL Server. In other words, you won't have to connect to a SQL Server to design a SSIS package. You can design the package disconnected from your SQL Server environment and then deploy it to your target SQL Server you'd like it to run on. This interface will be discussed in much more detail in Chapter 3.

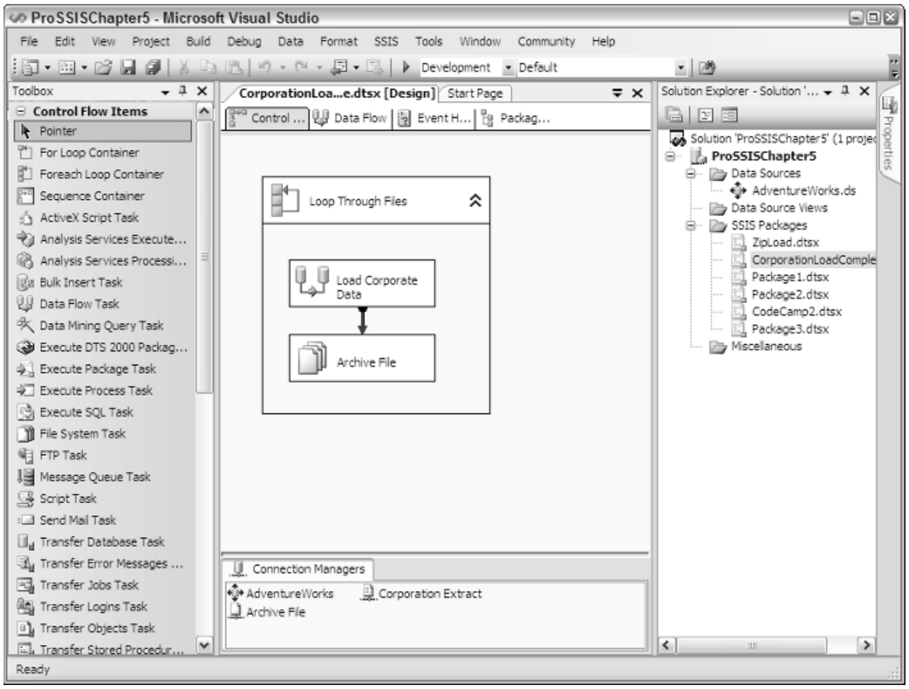


Figure 1-2

Architecture

SQL Server 2005 has truly evolved SSIS into a major player in the extraction, transformation, and loading (ETL) market. It was a complete code rewrite from SQL Server 2000 DTS. What's especially nice about SSIS is its price tag, which is free with the purchase of SQL Server. Other ETL tools can cost hundreds of thousands of dollars based on how you scale the software. The SSIS architecture has also expanded dramatically, as you can see in Figure 1-3. The SSIS architecture consists of four main components:

- The SSIS Service
- The SSIS runtime engine and the runtime executables
- The SSIS data flow engine and the data flow components
- The SSIS clients

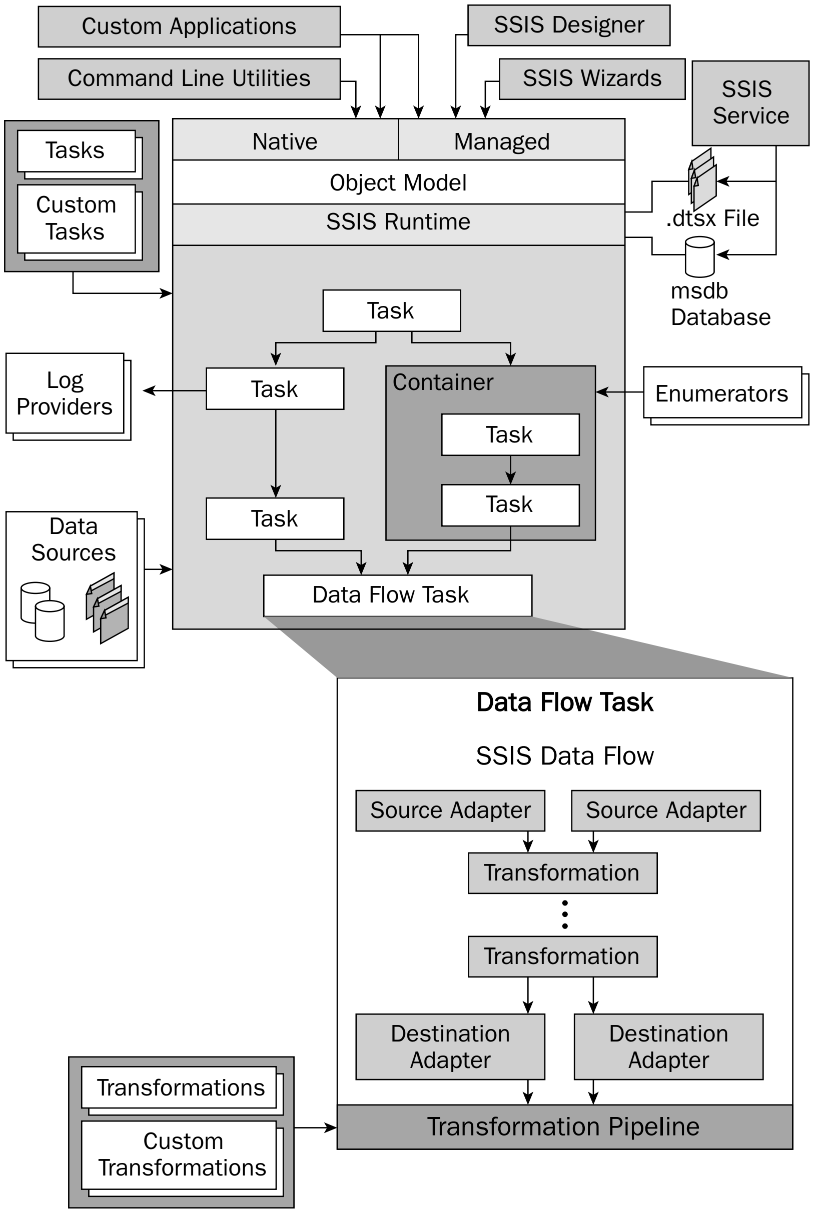


Figure 1-3

The SSIS Service handles the operational aspects of SSIS. It is a Windows service that is installed when you install the SSIS component of SQL Server 2005, and it tracks the execution of packages (a collection of work items) and helps with the storage of the packages. Don't worry; you'll learn more about what packages are momentarily. The SSIS Service is turned off by default and is set to disabled. It only turns on when a package is executed for the first time. You don't need the SSIS service to run SSIS packages, but if the service is stopped, all the SSIS packages that are currently running will in turn stop.

The SSIS runtime engine and its complementary programs actually run your SSIS packages. The engine saves the layout of your packages and manages the logging, debugging, configuration, connections, and transactions. Additionally, it manages handling your events when one is raised in your package. The runtime executables provide the following functionality to a package that you'll explore in more detail later in this chapter:

- **Containers:** Provide structure and scope to your package
- **Tasks:** Provide the functionality to your package
- **Event Handlers:** Respond to raised events in your package
- **Precedence Constraints:** Provide ordinal relationship between various items in your package

In Chapter 3, you'll spend a lot of time in each of these architecture sections, but the vital ones are introduced here.

Packages

A core component of SSIS and DTS is the notion of a *package*. A package best parallels an executable program in Windows. Essentially, a package is a collection of tasks that execute in an orderly fashion. Precedence constraints help manage which order the tasks will execute in. A package can be saved onto a SQL Server, which in actuality is saved in the msdb database. It can also be saved as a .DTSX file, which is an XML-structured file much like .RDL files are to Reporting Services. Of course, there is much more to packages than that, but you'll explore the other elements of packages, like event handlers, later in this chapter.

Tasks

A *task* can best be described as an individual unit of work. They provide functionality to your package, in much the same way that a method does in a programming language. The following are some of the tasks available to you:

- **ActiveX Script Task:** Executes an ActiveX script in your SSIS package. This task is mostly for legacy DTS packages.
- **Analysis Services Execute DDL Task:** Executes a DDL task in Analysis Services. For example, this can create, drop, or alter a cube.
- **Analysis Services Processing Task:** This task processes a SQL Server Analysis Services cube, dimension, or mining model.
- **Bulk Insert Task:** Loads data into a table by using the `BULK INSERT SQL` command.
- **Data Flow Task:** This very specialized task loads and transforms data into an OLE DB destination.
- **Data Mining Query Task:** Allows you to run predictive queries against your Analysis Services data-mining models.
- **Execute DTS 2000 Package Task:** Exposes legacy SQL Server 2000 DTS packages to your SSIS 2005 package.

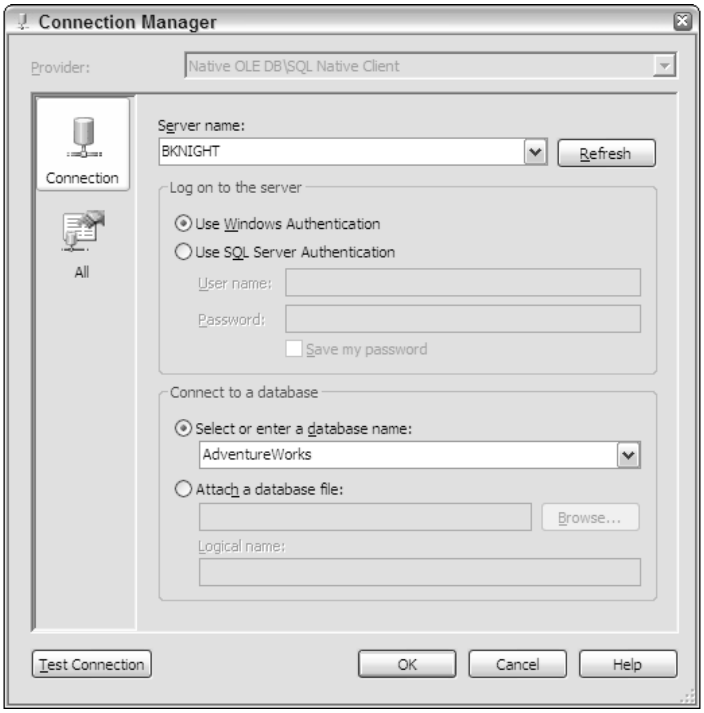
- **Execute Package Task:** Allows you to execute a package from within a package, making your SSIS packages modular.
- **Execute Process Task:** Executes a program external to your package, such as one to split your extract file into many files before processing the individual files.
- **Execute SQL Task:** Executes a SQL statement or stored procedure.
- **File System Task:** This task can handle directory operations such as creating, renaming, or deleting a directory. It can also manage file operations such as moving, copying, or deleting files.
- **FTP Task:** Sends or receives files from an FTP site.
- **Message Queue Task:** Send or receives messages from a Microsoft Message Queue (MSMQ).
- **Script Task:** Slightly more advanced than the ActiveX Script task. This task allows you to perform more intense scripting in the Visual Studio programming environment.
- **Send Mail Task:** Send a mail message through SMTP.
- **Web Service Task:** Executes a method on a Web service.
- **WMI Data Reader Task:** This task can run WQL queries against the Windows Management Instrumentation. This allows you to read the event log, get a list of applications that are installed, or determine hardware that is installed, to name a few examples.
- **WMI Event Watcher Task:** This task empowers SSIS to wait for and respond to certain WMI events that occur in the operating system.
- **XML Task:** Parses or processes an XML file. It can merge, split, or reformat an XML file.

There is also an array of tasks that can be used to maintain your SQL Server environment. These tasks perform functions such as transferring your SQL Server databases, backing up your database, or shrinking the database. Each of the tasks available to you is described in Chapter 3 in much more detail, and those tasks will be used in many examples throughout the book. Tasks are extensible, and you can create your own tasks in a language like C# to perform tasks in your environment, such as reading data from your proprietary mainframe.

Data Source Elements

The main purpose of SSIS remains lifting data, transforming it, and writing it to a destination. Data sources are the connections that can be used for the source or destination to transform that data. A data source can be nearly any OLE-DB-compliant data source such as SQL Server, Oracle, DB2, or even nontraditional data sources such as Analysis Services and Outlook. The data sources can be localized to a single SSIS package or shared across multiple packages in BIDS.

A connection is defined in the Connection Manager. The Connection Manager dialog box may vary vastly based on the type of connection you're trying to configure. Figure 1-4 shows you what a typical connection to SQL Server would look like.



Provider: Native OLE DB\SQL Native Client

Server name: BKNIGHT

(●) Use Windows Authentication
( ) Use SQL Server Authentication

User name:

Password:

[ ] Save my password

(●) Select or enter a database name: AdventureWorks
( ) Attach a database file:

Logical name:

Figure 1-4

You can configure the connection completely offline, and the SSIS package will not use it until you begin to instantiate it in the package. The nice thing about this is that you can develop in an airport and then connect as needed.

Data Source Views

Data source views (DSVs) are a new concept to SQL Server 2005. This feature allows you to create a logical view of your business data. They are a collection of tables, views, stored procedures, and queries that can be shared across your project and leveraged in Analysis Services and Report Builder.

This is especially useful in large complex data models that are prevalent in ERP systems like Siebel or SAP. These systems have column names like ER328F2 to make the data model flexible to support nearly any environment. This complex model naming convention creates positions of people in companies who specialize in just reading the model for reports. The business user, though, would never know what a column like this means, so a DSV may map this column to an entity like LastPaymentDate. It also maps the relationships between the tables that may not necessarily exist in the physical model.

DSVs also allow you to segment a large data model into more bite-sized chunks. For example, your Siebel system may be segmented into a DSV called Accounting, Human Resource, and Inventory. One example called Human Resource can be seen in Figure 1-5. As you can see in this figure, a friendly name has been assigned to one column called Birth Date (previously named BirthDate without the space) in the Employee entity. While this is a simplistic example, it's especially useful for the ER328F2 column previously mentioned.

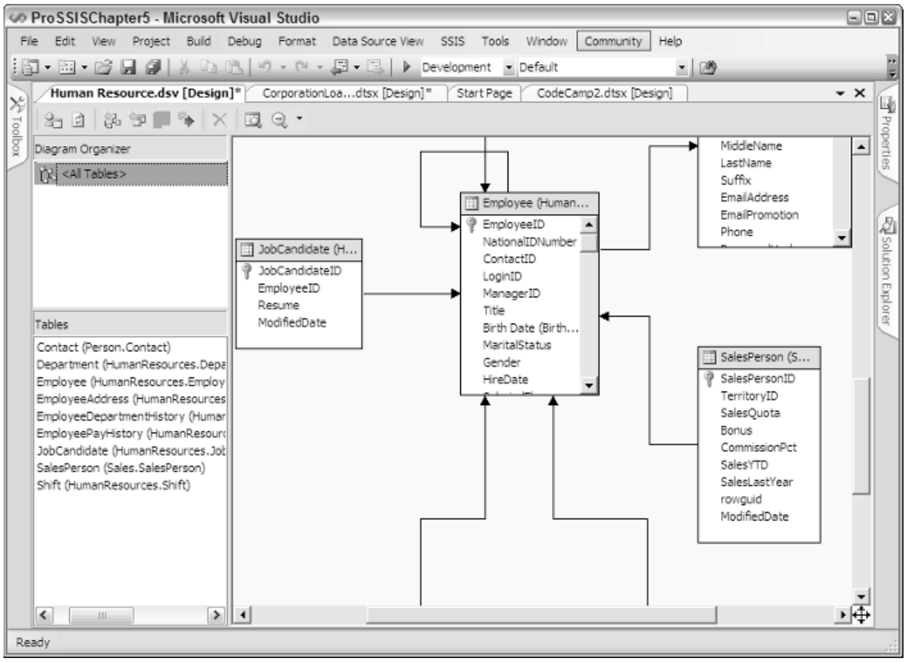


Figure 1-5

DSVs are deployed as a connection manager. There are a few key things to remember with data source views. Like data sources, DSVs allow you to define the connection logic once and reuse it across your SSIS packages. Unlike connections, though, DSVs are disconnected from the source connection and are not refreshed as the source structure changes. For example, if you change the Employee table in a connection to Resources, the DSV will not pick up the change. Where this type of caching is a huge benefit is in development. DSVs allow you to utilize cached metadata in development, even if you're in an airport, disconnected. It also speeds up package development. Since your DSV is most likely a subset of the actual data source, your SSIS connection dialog boxes will load much faster.

Precedence Constraints

Precedence constraints direct the tasks to execute in a given order. They direct the workflow of your SSIS package based on given conditions. Precedence constraints have been enhanced dramatically in SQL Server 2005 Integration Services conditional branching of your workflow based on conditions.

Constraint Value

Constraint values are the type of precedence constraint that you may be familiar with in SQL Server 2000. There are three types of constraint values:

- ❑ **Success:** A task that's chained to another task with this constraint will execute only if the prior task completes successfully.
- ❑ **Completion:** A task that's chained to another task with this constraint will execute if the prior task completes. Whether the prior task succeeds or fails is inconsequential.
- ❑ **Failure:** A task that's chained to another task with this constraint will execute only if the prior task fails to complete. This type of constraint is usually used to notify an operator of a failed event or write bad records to an exception queue.

Conditional Expressions

The nicest improvement to precedence constraints in SSIS 2005 is the ability to dynamically follow workflow paths based on certain conditions being met. These conditions use the new conditional expressions to drive the workflow. An *expression* allows you to evaluate whether certain conditions have been met before the task is executed and the path followed. The *constraint* evaluates only the success or failure of the previous task to determine whether the next step will be executed. The SSIS developer can set the conditions by using evaluation operators. Once you create a precedence constraint, you can set the EvalOp property to any one of the following options:

- ❑ **Constraint:** This is the default setting and specifies that only the constraint will be followed in the workflow.
- ❑ **Expression:** This option gives you the ability to write an expression (much like VB.NET) that allows you to control the workflow based on conditions that you specify.
- ❑ **ExpressionAndConstraint:** Specifies that both the expression and the constraint must be met before proceeding.
- ❑ **ExpressionOrConstraint:** Specifies that either the expression or the constraint can be met before proceeding.

An example workflow can be seen in Figure 1-6. This package first copies files using the File System task, and if that is successful and meets certain criteria in the expression, it will transform the files using the Data Flow task. If the first step fails, then a message will be sent to the user by using the Send Mail task. You can also see the small *fx* icon above the Data Flow task. This is graphically showing the developer that this task will not execute unless an expression has also been met and the previous step has successfully completed. The expression can check anything, such as looking at a checksum, before running the Data Flow task.

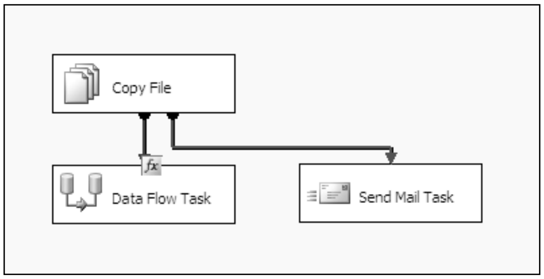


Figure 1-6

Containers

Containers are a new concept in SSIS that didn't previously exist in SQL Server. They are a core unit in the SSIS architecture that help you logically group tasks together into units of work or create complex conditions. By using containers, SSIS variables and event handlers (these will be discussed in a moment) can be defined to have the scope of the container instead of the package. There are four types of containers that can be employed in SSIS:

- ❑ **Task host container:** The core type of container that every task implicitly belongs to by default. The SSIS architecture extends variables and event handlers to the task through the task host container.
- ❑ **Sequence container:** Allows you to group tasks into logical subject areas. In BIDS, you can then collapse or expand this container for usability.
- ❑ **For loop container:** Loops through a series of tasks for a given amount of time or until a condition is met.
- ❑ **For each loop container:** Loops through a series of files or records in a data set and then executes the tasks in the container for each record in the collection.

As you read through this book, you'll gain lots of experience with the various types of containers.

Variables

Variables are one of the most powerful components of the SSIS architecture. In SQL Server 7.0 and 2000 DTS, these were called global variables, but they've been drastically improved on in SSIS. Variables allow you to dynamically configure a package at runtime. Without variables, each time you wanted to deploy a package from development to production, you'd have to open the package and change all the hard-coded connection settings to point to the new environment. Now with variables, you can just change the variables at deployment time, and anything that uses those variables will in turn be changed. Variables have the scope of an individual container, package, or system.

Data Flow Elements

Once you create a Data Flow task, it spawns a new data flow. Just as the Controller Flow handles the main workflow of the package, the data flow handles the transformation of data. Almost anything that manipulates data falls into the data flow category. As data moves through each step of the data flow, the data changes based on what the transform does. For example in Figure 1-7, a new column is derived using the Derived Column transform, and that new column is then available to subsequent transformations or to the destination.

In this section, each of the sources, destinations, and transformations will be briefly covered. These areas are covered in much more detail in Chapters 3 and 4.

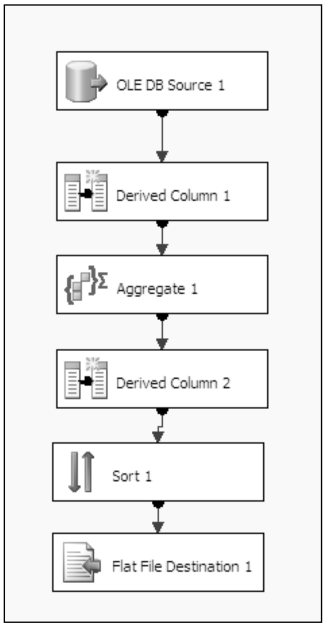


Figure 1-7

Sources

A *source* is where you specify the location of your source data to pull from in the data pump. Sources will generally point to the Connection Manager in SSIS. By pointing to the Connection Manager, you can reuse connections throughout your package, because you need only change the connection in one place. There are six sources altogether that can be used out of the box with SSIS:

- **OLE DB Source:** Connects to nearly any OLE DB data source, such as SQL Server, Access, Oracle, or DB2, to name just a few.
- **Excel Source:** Source that specializes in receiving data from Excel spreadsheets. This source also makes it easy to run SQL queries against your Excel spreadsheet to narrow the scope of the data that you wish to pass through the flow.
- **Flat File Source:** Connects to a delimited or fixed-width file.
- **Raw File Source:** A specialized file format that was produced by a Raw File Destination (discussed momentarily). The Raw File Source usually represents data that is in transit and is especially quick to read.
- **XML Source:** Can retrieve data from an XML document.
- **Data Reader Source:** The DataReader source is an ADO.NET connection much like the one you see in the .NET Framework when you use the DataReader interface in your application code to connect to a database.

Destinations

Inside the data flow, destinations accept the data from the data sources and from the transformations. The flexible architecture can send the data to nearly any OLE DB–compliant data source or to a flat file. Like sources, destinations are managed through the Connection Manager. The following destinations are available to you in SSIS:

- **Data Mining Model Training:** This destination trains an Analysis Services mining model by passing in data from the data flow to the destination.
- **DataReader Destination:** Allows you to expose data to other external processes, such as Reporting Services or your own .NET application. It uses the ADO.NET DataReader interface to do this.
- **Dimension Processing:** Loads and processes an Analysis Services dimension. It can perform a full, update, or incremental refresh of the dimension.
- **Excel Destination:** Outputs data from the data flow to an Excel spreadsheet.
- **Flat File Destination:** Enables you to write data to a comma-delimited or fixed-width file.
- **OLE DB Destination:** Outputs data to an OLE DB data connection like SQL Server, Oracle, or Access.
- **Partition Processing:** Enables you to perform incremental, full, or update processing of an Analysis Services partition.
- **Raw File Destination:** This destination outputs data that can later be used in the Raw File Source. It is a very specialized format that is very quick to output to.
- **Recordset Destination:** Writes the records to an ADO record set.
- **SQL Server Destination:** The destination that you use to write data to SQL Server most efficiently.
- **SQL Server Mobile Destination:** Inserts data into a SQL Server running on a Pocket PC.

Transformations

Transformations are key components to the data flow that change the data to a desired format. For example, you may want your data to be sorted and aggregated. Two transformations can accomplish this task for you. The nicest thing about transformations in SSIS is that it's all done in-memory and it no longer requires elaborate scripting as in SQL Server 2000 DTS. The transformation is covered in Chapters 4 and 6. Here's a complete list of transforms:

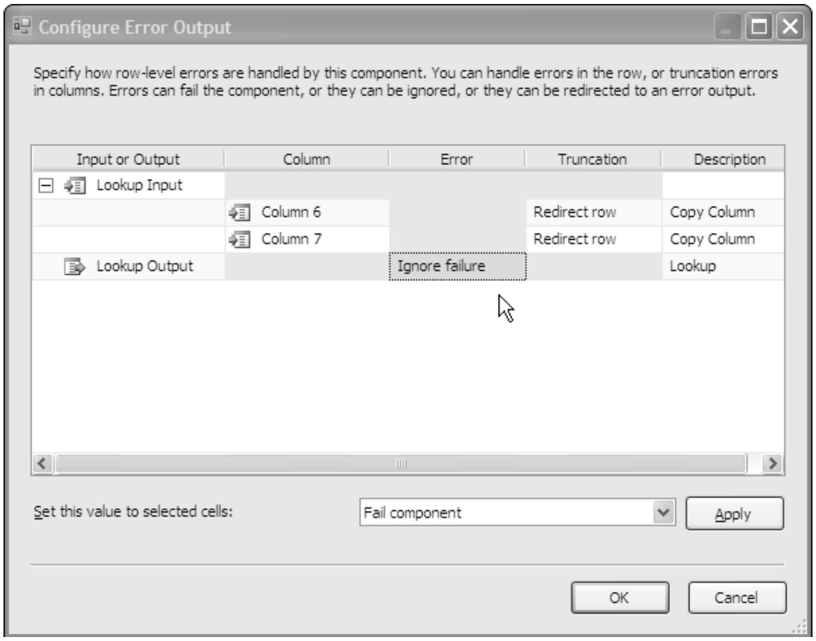
- ❑ **Aggregate:** Aggregates data from transform or source.
- ❑ **Audit:** The transformation that exposes auditing information to the package, such as when the package was run and by whom.
- ❑ **Character Map:** This transformation makes string data changes for you, such as changing data from lowercase to uppercase.
- ❑ **Conditional Split:** Splits the data based on certain conditions being met. For example, this transformation could be instructed to send data down a different path if the State column is equal to Florida.
- ❑ **Copy Column:** Adds a copy of a column to the transformation output. You can later transform the copy, keeping the original for auditing purposes.
- ❑ **Data Conversion:** Converts a column's data type to another data type.
- ❑ **Data Mining Query:** Performs a data-mining query against Analysis Services.
- ❑ **Derived Column:** Creates a new derived column calculated from an expression.
- ❑ **Export Column:** This transformation allows you to export a column from the data flow to a file. For example, you can use this transformation to write a column that contains an image to a file.
- ❑ **Fuzzy Grouping:** Performs data cleansing by finding rows that are likely duplicates.
- ❑ **Fuzzy Lookup:** Matches and standardizes data based on fuzzy logic. For example, this can transform the name Jon to John.
- ❑ **Import Column:** Reads data from a file and adds it into a data flow.
- ❑ **Lookup:** Performs a lookup on data to be used later in a transformation. For example, you can use this transformation to look up a city based on the zip code.
- ❑ **Merge:** Merges two sorted data sets into a single data set in a data flow.
- ❑ **Merge Join:** Merges two data sets into a single data set using a join function.
- ❑ **Multicast:** Sends a copy of the data to an additional path in the workflow.
- ❑ **OLE DB Command:** Executes an OLE DB command for each row in the data flow.
- ❑ **Percentage Sampling:** Captures a sampling of the data from the data flow by using a percentage of the total rows in the data flow.
- ❑ **Pivot:** Pivots the data on a column into a more non-relational form. *Pivoting* a table means that you can slice the data in multiple ways, much like in OLAP and Excel.

- **Row Count:** Stores the row count from the data flow into a variable.
- **Row Sampling:** Captures a sampling of the data from the data flow by using a row count of the total rows in the data flow.
- **Script Component:** Uses a script to transform the data. For example, you can use this to apply specialized business logic to your data flow.
- **Slowly Changing Dimension:** Coordinates the conditional insert or update of data in a slowly changing dimension. You'll learn the definition of this term and study the process in Chapter 6.
- **Sort:** Sorts the data in the data flow by a given column.
- **Term Extraction:** Looks up a noun or adjective in text data.
- **Term Lookup:** Looks up terms extracted from text and references the value from a reference table.
- **Union All:** Merges multiple data sets into a single data set.
- **Unpivot:** Unpivots the data from a non-normalized format to a relational format.

Error Handling and Logging

In SSIS, the package events are exposed in the user interface, with each event having the possibility of its own event handler design surface. This *design surface* is the pane in Visual Studio where you can specify a series of tasks to be performed if a given event happens. There are a multitude of event handlers to help you develop packages that can self-fix problems. For example, the OnError error handler triggers an event whenever an error occurs anywhere in scope. The scope can be the entire package or an individual container. Event handlers are represented as a workflow, much like any other workflow in SSIS. An ideal use for event handlers would be to notify an operator if any component fails inside the package. You'll learn much more about event handlers in Chapter 13.

Handling errors in your data is easy now in SSIS 2005. In the data flow, you can specify in a transformation or connection what you wish to happen if an error exists in your data. You can select that the entire transformation fails and exits upon an error, or the bad rows can be redirected to a failed data flow branch. You can also choose to ignore any errors. An example of an error handler can be seen in Figure 1-8, where if an error occurs during the Derived Column transformation, it will be outputted to the data flow. You can then use that outputted information to write to an output log.



Specify how row-level errors are handled by this component. You can handle errors in the row, or truncation errors in columns. Errors can fail the component, or they can be ignored, or they can be redirected to an error output.

Input or Output	Column	Error	Truncation	Description
Lookup Input				
	Column 6		Redirect row	Copy Column
	Column 7		Redirect row	Copy Column
Lookup Output		Ignore failure		Lookup

Set this value to selected cells: Fail component

Figure 1-8

Once configured, you can specify that the bad records be written to another connection, as shown in Figure 1-9. The On Failure precedence constraint can be seen as a red line that connects the Derived Column 1 task to the SQL Server Destination. The green arrows are the On Success precedence constraints. You can see the On Success constraint between the OLE DB Source and the Derived Column transform.

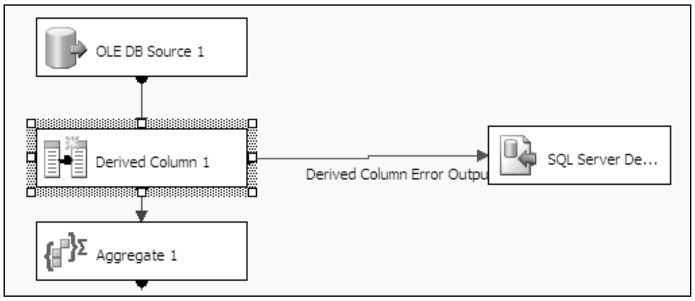


Figure 1-9

Logging has also been improved in SSIS 2005. It is now at a much finer detail than in SQL Server 2000 DTS. There are more than a dozen events that can be logged for each task or package. You can enable partial logging for one task and enable much more detailed logging for billing tasks. Some of the events that can be monitored are OnError, OnPostValidate, OnProgress, and OnWarning, to name just a few. The logs can be written to nearly any connection: SQL Profiler, text files, SQL Server, the Windows Event log, or an XML file.

Editions of SQL Server 2005

The available features in SSIS and SQL Server vary widely based on what edition of SQL Server you're using. As you can imagine, the more high-end the edition of SQL Server, the more features are available. In order from high-end to low-end, the following SQL Server editions are available:

- **SQL Server 2005 Enterprise Edition:** The edition of SQL Server for large enterprises that need higher availability and more advanced features in SQL Server and business intelligence. For example, there is no limit on processors or RAM in this edition. You're bound only by the number of processors and amount of RAM that the OS can handle. This edition is available for an estimated retail price (ERP) of $24,999 (U.S.) per processor or $13,499 (U.S.) per server (25 CALs). Microsoft will also continue to support Developer Edition, which lets developers develop SQL Server solutions at a much reduced price. That edition has all the features of Enterprise Edition but is licensed for development purposes only.
- **SQL Server 2005 Standard Edition:** This edition of SQL Server has a lot more value in SQL Server 2005. For example, you can now create a highly available system in Standard Edition by using clustering, database mirroring, and integrated 64-bit support. These features were available only in Enterprise Edition in SQL Server 2000 and caused many businesses to purchase Enterprise Edition when Standard Edition was probably sufficient for them. Like Enterprise Edition in SQL Server 2005, it also offers unlimited RAM! Thus, you can scale it as high as your physical hardware and OS will allow. There is a cap of four processors, though. Standard Edition is available for an ERP of $5,999 (U.S.) per processor or $2,799 (U.S.) per server (10 CALs).
- **SQL Server 2000 and 2005 Workgroup Editions:** This new edition is designed for small and medium-sized businesses that need a database server with limited business intelligence and Reporting Services. Available for an ERP of $3,899 (U.S.) per processor or $739 (U.S.) per server (5 CALs), Workgroup Edition supports up to two processors with unlimited database size. In SQL Server 2000 Workgroup Edition, the limit is 2 GB of RAM. In SQL Server 2005 Workgroup Edition, the memory limit has been raised to 3 GB.
- **SQL Server 2005 Express Edition:** This edition is the equivalent of Desktop Edition (MSDE) in SQL Server 2000 but with several enhancements. For example, MSDE never offered any type of management tool, and this is included in 2005. Also included are the Import and Export Wizard and a series of other enhancements. This remains a free addition of SQL Server for small applications. It has a database size limit of 4 GB. Most important, the query governor has been removed from this edition, allowing for more people to query the instance at the same time.

As for SSIS, you'll have to use at least Standard Edition to receive the bulk of the SSIS features. In the Express and Workgroup Editions, only the Import and Export Wizard is available to you. You'll have to upgrade to Enterprise or Developer Edition to see some features in SSIS. The following advanced transformations are available only with Enterprise Edition:

- Analysis Services Partition Processing Destination
- Analysis Services Dimension Processing Destination
- Data Mining Training Destination
- Data Mining Query Component
- Fuzzy Grouping
- Fuzzy Lookup
- Term Extraction
- Term Lookup

Half of the above transformations are used in servicing Analysis Services. To continue that theme, one task is available only in Enterprise Edition — the Data Mining Query task.

Summary

In this chapter, you were introduced to the SQL Server Integration Services (SSIS) architecture and some of the different elements you'll be dealing with in SSIS. Tasks are individual units of work that are chained together with precedence constraints. Packages are executable programs in SSIS that are a collection of tasks. Lastly, transformations are the data flow items that change the data to the form you request, such as sorting the data.

In Chapter 2, you'll study some of the wizards you have at your disposal to expedite tasks in SSIS, and in Chapter 3, you'll dive deeper into the various SSIS tasks.